J Heuristics (2016) 22:699–726
DOI 10.1007/s10732-016-9316-7

A novel GRASP solution approach for the Orienteering Problem

Morteza Keshtkaran[1] · **Koorush Ziarati**[1]

Received: 28 March 2016 / Revised: 15 June 2016 / Accepted: 9 September 2016 /
Published online: 17 September 2016

Abstract The Orienteering Problem (OP) is a well-known variant of the Traveling Salesman Problem. In this paper, a novel Greedy Randomized Adaptive Search Procedure (GRASP) solution is proposed to solve the OP. The proposed method is shown to outperform state-of-the-art heuristics for the OP in producing high quality solutions. In comparison with the best known solutions of standard benchmark instances, the method can find the optimal or the best known solution of about 70 % of the instances in a reasonable time, which is about 17 % better than the best known approach in the literature. Moreover, a significant improvement is achieved on the solution of two standard benchmark instances.

Keywords Traveling Salesman Problem · Orienteering Problem · Heuristic · GRASP

1 Introduction

The Orienteering Problem (OP) is a well-known variant of the Traveling Salesman Problem (TSP). The problem is inspired from an outdoor game usually played in mountainous or forested areas, where a set of checkpoints (landmarks) is available. Each checkpoint is associated with a profit and can be visited at most once. Within a predefined time limit, a contestant starts from the origin, visits a subset of the checkpoints, and finishes at the destination. Since it may not be possible to visit all the

✉ Morteza Keshtkaran
mkeshtkaran@cse.shirazu.ac.ir

Koorush Ziarati
ziarati@shirazu.ac.ir

1 Department of Electrical and Computer Engineering, Shiraz University, Shiraz, Iran

checkpoints, the objective in the OP is to collect the maximum possible profit by visiting some preferred checkpoints.

The OP is an NP-hard problem (Laporte and Martello 1990), which was originally introduced by Tsiligirides (1984). This problem arises in many real-world applications, including athlete recruiting (Chao et al. 1996b), technician routing (Tang and Miller-Hooks 2005), and tourist trip planning (Vansteenwegen et al. 2009).

Due to its NP-hardness, the branch-and-bound (Laporte and Martello 1990; Ramesh et al. 1992) and branch-and-cut (Fischetti et al. 1998; Gendreau et al. 1998a) exact algorithms, which are proposed to solve the problem to optimality, are usually time consuming. Therefore, most researches have focused on heuristic approaches. Among these heuristics, the most successful approaches, which have been recently proposed, include the Tabu Search heuristic (TS) (Gendreau et al. 1998a), Ant Colony Optimization (ACO) (Schilde et al. 2009), 2-Parameter Iterative Algorithm (2-PIA) (Silberholz and Golden 2010), and GRASP[1] with Path Relinking (GRASP-PR) (Campos et al. 2014). For a comprehensive review of the OP and its solutions, the reader is referred to the recent surveys by Vansteenwegen et al. (2011), Archetti et al. (2014), and Gavalas et al. (2014).

GRASP is a metaheuristic algorithm commonly applied to combinatorial optimization problems. In this algorithm, successive constructions of greedy randomized solutions are followed by iterative improvements through a local search. In this paper, we reconsider the GRASP solution approach introduced by Campos et al. (2014) to develop a novel GRASP heuristic method for the OP, which is shown to be competitive with the state-of-the-art.

The paper is organized as follows. The formal definition of the problem and one of its mathematical integer linear formulations are described in Sect. 2. Section 3 is devoted to the description of the proposed GRASP method for solving the OP. Various experimental results are presented in Sect. 4, the paper ends with some concluding remarks in Sect. 5.

2 Problem description and formulation

The OP can be modeled on a complete graph $G = (V, E)$ where $V = \{0, 1, \cdots, n, n+1\}$ is the set of vertices in G, and $E = \{(u, v)|u, v \in V\}$ represents the set of edges. Vertices 0 and $n+1$ are the origin and the destination points, respectively, while vertices 1 to n are the potential checkpoints. $V_0 = V \backslash \{0, n+1\}$ represents the checkpoint set. A travel time, t_{uv}, between each pair of vertices $(u, v) \in E$ is given, and a profit p_v is assigned to each vertex $v \in V$.

A solution to the problem is a path that begins from vertex 0, visit a subset of vertices in V_0, and ends at vertex $n+1$. Each vertex in the subset must be visited at most once, and the time taken to visit the vertices of the solution cannot exceed T_{max}. The aim of the OP is to collect the maximum profit from the visited vertices.

For a formal definition, the following integer programming formulation of the problem has been presented (Vansteenwegen et al. 2011).

[1] Greedy Randomized Adaptive Search Procedure (Feo and Resende 1995).

Maximize

$$\sum_{i=1}^{n}\sum_{j=1}^{n+1} p_i x_{ij}, \tag{1}$$

Subject to

$$\sum_{i=1}^{n+1} x_{0i} = \sum_{i=0}^{n} x_{i(n+1)} = 1, \tag{2}$$

$$\sum_{i=0}^{n} x_{iv} = \sum_{j=1}^{n+1} x_{vj} \leq 1; \qquad v \in \{1, 2, \cdots, n\}, \tag{3}$$

$$\sum_{i=0}^{n}\sum_{j=1}^{n+1} t_{ij} x_{ij} \leq T_{max}, \tag{4}$$

$$1 \leq u_i \leq n; \qquad i \in \{1, 2, \cdots, n\}, \tag{5}$$

$$u_i - u_j + 1 \leq (1 - x_{ij})n; \qquad i, j \in \{1, 2, \cdots, n\}, \tag{6}$$

$$x_{ij} \in \{0, 1\}; \qquad i, j \in \{1, 2, \cdots, n\}, \tag{7}$$

$$u_i \in \mathbb{Z}^+; \qquad i \in \{1, 2, \cdots, n\}. \tag{8}$$

In this formulation, two sets of decision variables x and u are available. $x_{ij} = 1$ if a visit to vertex i is followed by a visit to vertex j, and 0 otherwise; u_i is equal to the position of vertex i in the solution path.

The objective function (1) is to maximize the total profit of the visited vertices. Constraint 2 ensures that the path starts at vertex 0 and ends at vertex $n+1$. Constraint 3 ensures that the path is connected and each vertex is visited at most once. Constraint 4 ensures that the path meets the time budget. Finally, Constraints 5 and 6 ensure that there are no subtours.

3 A novel GRASP heuristic for solving the OP

GRASP (Feo and Resende 1995) is a multi-start metaheuristic commonly used for solving combinatorial optimization problems. In each iteration, a solution is generated by applying two phases: a construction and a local search. The best solution of all iterations is kept as the result.

The construction phase basically starts with an empty solution as its first partial solution. For each partial solution, a candidate list of elements that can extend the partial solution to another feasible solution is created. This list is then restricted to more eligible candidates by an evaluation function. Next, a random candidate is selected from the list and the partial solution is extended by that candidate. The candidate list is then updated for the new partial solution and when there are no candidate elements that can extend the last constructed partial solution, the construction phase stops. Finally, the quality of the constructed solution is improved through a local search.

Algorithm 1 GRASP-SR

```
function GRASP- SR()
                                                          ▷ Construction phase
    curPath = bestPath =< 0, n + 1 >
    while AddVertex() do                ▷ Note: This loop does not have any statements.
                                                          ▷ Local search phase
    bestPath = curPath
    while LocalSearch() do              ▷ Note: This loop does not have any statements.

    return bestPath


function AddVertex(blocked = −1)
                           ▷ Add a new vertex to the curPath – O((n − l)l + l^2)
    r = curPath
    l = |curPath| − 1;
    CL = {}

    for w := 1 to n do                                           ▷ O((n − l)l)
        if w ∉ r and w ≠ blocked then
            i = arg min_{1≤k≤l} {T_{r'} | r' =< r[0], ..., r[k − 1], w, r[k], ..., r[l] >}   ▷ O(l)
            r' =< r[0], ..., r[i − 1], w, r[i], ..., r[l] >}
            if T_{r'} ≤ T_max then
                CL = CL ∪ {r'}
            else                                                 ▷ O(l)
                j = 1
                for i := 1 to l do
                    if j < i then
                        j = i
                    while j ≠ l and T_{r''=<r'[0],...,r'[i−1],r'[j+1],...,r'[l+1]>} > T_max do
                        j = j + 1
                    r'' =< r'[0], ..., r'[i − 1], r'[j + 1], ..., r'[l + 1] >
                    if T_{r''} ≤ T_max then
                        if P_{r''} > P_r or (P_{r''} = P_r and T_{r''} < T_r) then
                            CL = CL ∪ {r''}
    if CL ≠ {} then                                              ▷ O(l^2)
        best = Max{P_i − P_r | i ∈ CL}
        RCL = {i | i ∈ CL and P_i − P_r ≥ 0.2 × best}
        curPath = a random path from RCL after being improved by 2-Opt
        return True
    return False

function LocalSearch()
    r = bestPath
    l = |bestPath| − 2;
    improved = False
    for i := 1 to l do
        curPath =< r[0], ..., r[i − 1], r[i + 1], ..., r[l + 1] >
        Apply 2-opt to curPath
        while AddVertex(r[i]) do        ▷ Note: This loop does not have any statements.
        if P_curPath > P_bestPath or (P_curPath = P_bestPath and T_curPath < T_bestPath) then
            bestPath = curPath
            improved = True
    return improved
```

Campos et al. (2014) proposed a GRASP with path relinking to solve the OP. In this section, we reconsider their work and propose a novel GRASP heuristic for the OP, which is competitive with GRASP using path relinking and also competitive with the state-of-the-art. In what follows the proposed heuristic approach is denoted by GRASP-SR (GRASP with Segment Remove).

In this section, we represent each OP solution path r as a sequence of vertices $< r[0] = 0, r[1], \cdots, r[l] = n+1 >$, where $l+1$ is the number of visited vertices of the graph, denoted by $|r|$. In addition, the total profit and travel time of path r are denoted by P_r and T_r, respectively.

The GRASP-SR algorithm is demonstrated in Algorithm 1. This algorithm consists of a construction phase (Lines 2–3) followed by a local search (Line 5), which are described in the following two subsections.

3.1 Our GRASP construction method

Our construction method starts with the path $r =< r[0] = 0, r[l] = n+1 >$ $(l = |r|-1)$, which goes directly from vertex 0 to vertex $n+1$. This path is considered as the current path and we try to improve its quality by successive insertion of new vertices into the path. New vertices are added by function $AddVertex()$ as follows.

Consider CL as the candidate list of paths, which are better in quality than the current path. At the beginning, this list is empty (Line 12). Each vertex $w \notin r$ is inserted in the best position of r (i.e., the one that produces the minimum path travel time) without changing the relative order of the vertices in the current path (Lines 16–17). If the new path r' is feasible, it is added to the candidate list (Lines 18–19); otherwise, for each position $0 < i < l+1$ in $r' =< r'[0], \cdots, r'[i], \cdots, r'[j], \cdots, r'[l+1] >$, the position $i \leq j < l+1$ is found (if possible), such that $j-i$ is minimal and the path $r'' =< r'[0], \cdots, r'[i-1], r'[j+1], \cdots, r'[l+1] >$ obtained by removing the vertices from position i to j of r' is feasible. [2] If $P_{r''} > P_r$, r'' is a better solution than r, and it is added to the candidate list, but if $P_{r''} = P_r$, r'' is added to the candidate list if its travel time is shorter than the travel time of r (Lines 21–30). The total time complexity of constructing the candidate list is $O((n-l)l)$.

If no improvement is possible, the CL becomes empty and the function returns $False$, showing no improvement is possible; otherwise, the maximum profit improvement of the paths is considered among the candidate list. The candidate list is restricted to the paths having an improvement within the fraction $\alpha = 0.2$ of this value. A discussion about the adjustment of this parameter is presented in Appendix 1.

$$best = Max\{P_i - P_r | i \in CL\} \tag{9}$$

$$RCL = \{i | i \in CL \text{ and } P_i - P_r \geq 0.2 \times best\} \tag{10}$$

Next, an element of the RCL is randomly selected and considered as the new current path. To reduce the length of this path, the 2-opt improvement mechanism is applied to the path and then the function returns $True$ (Lines 31–36).

[2] If the travel distance time matrix satisfies the triangular inequality, this can be done in time complexity $O(l)$. The algorithm is similar to the linear algorithm for solving the well-known "the smallest sub-array with sum greater than a given value" problem. Using the same algorithm, one can find j values for all positions of i (Lines 21–30). This $O(l)$ algorithm is applied even if the triangular inequality condition does not hold, sacrificing some quality.

The process of adding new vertices by calling $AddVertex()$ is repeated until no improvement is possible (Lines 3–4). The total time complexity of each improvement is $O((n-l)l + l^2)$, which is proportional to the number of vertices in the path.

The proposed construction method is similar to that of Campos et al. (2014) with the difference that when it is not possible to directly insert a vertex into the path, we try to remove a segment of the path to make space for the insertion of the new vertex. Thus, the proposed construction method is more general than the construction method of Campos et al. (2014) with the same overall time complexity.

3.2 Local search

The construction method of Campos et al. (2014) is followed by a two-phase local search. The first phase is based on some sequential one-to-one exchanges between the vertices in the path and the not less profitable vertices outside the path. Since these one-to-one exchanges may make space for the insertion of some new vertices into the path, the second phase is devoted to the insertion of these new vertices.

The proposed construction method contains both aforementioned local searches in its process. One-to-one exchanges is equivalent to the insertion of a vertex along with the removal of a segment of length one and the second phase is equivalent to the insertion of a vertex without any segment removal.

We have implemented another simple local search to improve the constructed path. This local search is applied by the $LocalSearch()$ function, which works as follows.

For each vertex $w \in r \setminus \{0, n+1\}$, remove the vertex from the path and apply the 2-opt optimization to the path. The resulting path is then improved by the construction method. In the construction method, w is excluded from the improved path (Line 15). If the best path constructed by this way is better than the original path, it is selected as the next path and the function returns $True$; otherwise, the function returns $False$, showing that no improvement is possible by the local search. This process is continued until no further improvement is possible.

Although this final local search increases the total time complexity of each improvement to $O((n-l)l^2+l^3)$, our experimental results show that GRASP-SR is competitive with the state-of-the-art in terms of both performance and run-time.

3.3 Example

To clarify the proposed algorithm, the algorithm is traced on the first instance of the p64 benchmark instances described in Sect. 4.1. The coordinates and the profits of the vertices required for following the example are given in Table 1. In this example, T_{max} is 15 and vertices 0 and 1 are the starting and the finishing points, respectively.

Since the proposed algorithm is a randomized algorithm, each run of the program may produce a different result. The following is an example run of the algorithm on this problem. At first, the paths constructed by inserting vertices 9, 41, 3 and 53 in their best positions are randomly selected in sequence. The constructed feasible path after these insertions is

Table 1 The coordinates and the profits of the required vertices of the first instance of p64 benchmark instances

Vertex	0	1	3	5	9	19	33	41	47	53	57	60
X	0	0	1	0	1	1	1	2	1	2	1	0
Y	−7	7	−6	−5	−4	−2	0	1	2	3	4	5
Profit	0	0	6	6	12	18	24	24	18	18	12	6

$$< 0, \underline{3, 9, 41, 53}, 1 > \qquad (T = 14.99, P = 60).$$

Inserting vertex 33 in its best position, results in an infeasible path. However, removing the segment containing vertices 3 and 9, makes the path feasible and still more profitable. This path is the next randomly selected path from the candidates list pool.

$$< 0 \cancel{, 3, 9}, \underline{33}, 41, 53, 1 > \qquad (T = 14.96, P = 66)$$

Similarly, vertices 5 and 19 are added to the path as follows.

$$< 0, \underline{5}, 33, 41, 53, 1 > \qquad (T = 14.99, P = 72)$$
$$< 0 \cancel{, 5}, \underline{19}, 33, 41, 53, 1 > \qquad (T = 14.99, P = 84)$$

It is seen that the insertion of vertex 5 is immediately improved by its replacement with vertex 19, which is more profitable than vertex 5. Since there is not any new profitable insertion with segment removal, the construction phase is finished.

Next, vertices 19, 33, 41 and 53 are independently removed from the constructed path and the possibility of new profitable insertions is checked for each of them. For this run of the program, removing vertex 41 produced the best result as follows.

$$< 0, 19, 33 \cancel{, 41}, 53, 1 > \qquad (T = 14.73, P = 60)$$
$$< 0, \underline{5}, 19, 33, 53, \underline{57}, 1 > \qquad (T = 14.90, P = 78)$$
$$< 0 \cancel{, 5}, \underline{9}, 19, 33, 53, 57, 1 > \qquad (T = 14.90, P = 84)$$
$$< 0, 9, 19, 33, \underline{47} \cancel{, 53}, 57, 1 > \qquad (T = 14.32, P = 84)$$
$$< 0, \underline{5}, 9, 19, 33, 47, 57, \underline{60}, 1 > \qquad (T = 14.83, P = 96)$$

This path cannot be further improved by the local search. As the results in Table 2 show, this is the best solution path for this instance.

4 Experimental results

In this section, the experimental results on OP standard benchmark instances are presented. GRASP-SR was implemented in C++ and tests were run on an Intel Core i7 with a 3.4 GHz CPU.

Two configurations of GRASP-SR are used in subsequent sections. In the first configuration, GRASP-SR is replicated 10 times on each instance. In each replication, 500 solutions are generated and the best of these solutions is returned. Therefore, in this configuration we have 5000 iterations to generate 5000 solutions. The output of our method for this configuration is described as follows: [3]

Best^n	The best solution and the number of times it was obtained.
Avg.	The average of the solutions.
Worst^n	The worst solution and the number of times it was obtained.
Sec.	The time in seconds in which the best solution was found for the first time.
#Iter.	The iteration number in which the best solution was found for the first time.
size	The number of visited vertices in the best solution path.

For the second configuration, the best solution obtained during the time limit of 2 min is considered. Column 2-min. corresponds to this configuration in the result tables. This configuration is used to show that our approach is competitive with the state-of-the-art, even if the time is restricted.

The first benchmark instances for this problem was proposed by Tsiligirides (1984), which consists of instances with the number of vertices ranging from 21 to 33. Since most of the current methods are able to solve these small instances to optimality, the corresponding results are not presented.

GRASP-SR was tested versus the best methods proposed in the literature on two standard benchmark problem sets. The results are presented in the following subsections.

4.1 Experimental results on p64 and p66 instances

The problems in the first set are graphs of size 64 and 66 vertices proposed in Chao et al. (1996a), called p64 and p66 instances, respectively. The two methods presented the best results for these instances are the ACO (Schilde et al. 2009) and GRASP with Path Relinking (GRASP-PR) (Campos et al. 2014), which were run on a Pentium 4D with a 3.2 GHz CPU and an Intel Core i5 with a 3.2 GHz CPU, respectively.

The results comparing our method with these methods on the p64 and the p66 instances are presented in Tables 2 and 3. The ACO, GRASP-PR, and there hereby proposed GRASP-SR method were repeated ten times on each instance, and for each method, the best and the worst results of the 10 runs (and only the best if both are the same), and the time in which the best result of the 10 runs was found are reported in columns Value and Sec., respectively.

Comparing the results of GRASP-SR with the ACO and GRASP-PR shows that GRASP-SR is the only method that was able to solve all the p64 instances to the best known solutions in all its runs. On the other hand, Table 3 shows that the ACO and

[3] The values for **Sec.**, **#iter**, and **size** are reported for the best solution among the 5,000 generated solutions, while the other outputs are reported according to the best result of each replication.

Table 2 GRASP-SR in comparison with the ACO and GRASP-PR approaches on p64 instances

T_{max}	ACO		GRASP-PR		GRASP-SR			
	Value	Sec.	Value	Sec.	Value	Sec.	$\#iter$	Size
15	96	0.007	96	0.015	96	0.000	1	9
20	294	0.017	294	0.062	294	0.000	1	15
25	390	0.025	390	0.14	390	0.000	1	18
30	474	0.034	468	0.171	474	0.031	14	20
35	576/570	0.508	576	0.28	576	0.015	5	24
40	714	0.409	714	0.28	714	0.171	78	29
45	816/804	7.013	816	0.296	816	0.062	10	32
50	900/894	4.492	900	0.421	900	0.016	3	35
55	984/978	8.323	984/978	0.296	984	0.000	2	39
60	1062/1056	0.991	1062/1044	0.249	1062	0.031	6	43
65	1116	0.711	1116	0.202	1116	0.000	1	46
70	1188	2.281	1188	0.187	1188	0.000	2	50
75	1236	0.721	1236	0.171	1236	0.031	10	53
80	1284/1278	2.109	1284/1278	0.14	1284	0.686	144	57

GRASP-PR generated the best results on the p66 instances. Although there are two instances for which our method did not produce the best result in all its runs, the best result was obtained, in at least half of the runs.

The computation times needed for solving these instances show that all the three methods can easily find the best results for these instances and are competitive with each other.

4.2 Experimental results on TSP-based benchmark instances

A larger set of benchmark instances was proposed in Fischetti et al. (1998) based on the TSPLIB 2.1 instances (Reinelt 1991). In this set, the TSP problems with up to 400 vertices are considered and converted to OP instances as follows. For each instance, the distance time limit is selected as $\lfloor \frac{Opt(P)}{2} \rfloor$, where Opt(P) is the length of the shortest Hamiltonian tour for the problem. The first vertex is considered as the origin and destination point (vertex $0 =$ vertex $n + 1$), and then according to the following rules each vertex of the TSP instance is assigned a profit value.

Generation 1:	$p_i = 1$	$i = 0, .., n,\ p_{n+1} = 0$
Generation 2:	$p_i = [1 + (7141 \times i + 73)]\ mod\ (100)$	$i = 0, .., n,\ p_{n+1} = 0$
Generation 3:	$p_i = 1 + \lfloor \frac{99 \times t_{0i}}{\theta} \rfloor$	$i = 1, .., n,\ p_0 = p_{n+1} = 0,$ $\theta = \max_{v \in V_0} t_{0v}$

Table 3 GRASP-SR in comparison with the ACO and GRASP-PR approaches on p66 instances

T_{max}	ACO		GRASP-PR		GRASP-SR				
	Value	Sec.	Value	Sec.	Best^n	Worst^n	Sec.	#iter	Size
5	10	0.01	10	0.00	10		0.00	1	4
10	40	0.01	40	0.00	40		0.00	1	6
15	120	0.01	120	0.02	120		0.00	4	8
20	205	0.06	205	0.02	205		0.00	2	11
25	290	0.02	290	0.03	290		0.02	1	12
30	400	0.02	400	0.05	400		0.02	1	16
35	465	0.03	465	0.06	465		0.02	4	19
40	575	0.04	575	0.28	575		0.02	1	21
45	650	0.05	650	0.08	650		0.00	1	24
50	730	0.05	730	0.09	730		0.00	1	26
55	825	0.06	825	0.09	825		0.00	2	29
60	915	0.13	915	0.11	915		0.02	2	31
65	980	0.08	980	0.11	980		0.00	1	34
70	1070	0.08	1070	0.11	1070		0.02	1	36
75	1140	0.09	1140	0.11	1140		0.00	1	38
80	1215	1.27	1215	0.12	1215		0.00	1	41
85	1270	0.22	1270	0.12	1270		0.00	1	44
90	1340	0.48	1340	0.12	1340		0.02	3	46
95	1395	0.39	1395	0.11	1395		0.02	1	49
100	1465	1.41	1465	0.41	1465		0.06	7	51
105	1520	0.15	1520	0.51	1520^6	1515^4	0.62	83	54
110	1560	0.32	1560	0.09	1560		0.39	62	56
115	1595	0.16	1595	0.09	1595		0.02	3	59
120	1635	1.22	1635	0.22	1635		0.20	55	61
125	1670	0.19	1670	0.19	1670^5	1665^5	0.25	69	64
130	1680	0.19	1680	0.02	1680		0.12	61	66

To the best of the authors' knowledge, there are only three methods that have been tested on these benchmark instances. Silberholz and Golden (2010) reported the results for the Tabu Search heuristic (TS) proposed by Gendreau et al. (1998b) and for their 2-Parameter Iterative Algorithm (2-PIA), running them on an Intel Pentium 4D with a 3.2 GHz CPU.

The third method is GRASP-PR (Campos et al. 2014), which used a different distance function and also different rules for generating the profits for these benchmark instances in comparison with Fischetti et al. (1998). According to our experimental results, the differences are as follows: Both Fischetti et al. (1998) and Campos et al. (2014) used the time distances t_{ij} as described in Reinelt (1991), but with different approaches to cast the real returned values to an integer. Campos et al. (2014) truncated the values while Fischetti et al. (1998) rounded the values. Additionally, the t_{ij} values used for computing the profit values in Generation 3 of Campos et al. (2014) were not casted to integer values. According to the mentioned time distance evaluations, the following rules were used in Campos et al. (2014) to generate the profits for the benchmark instances.

Generation 1:	$p_i = 1$	$i = 0, .., n,\ p_{n+1} = 0$
Generation 2:	$p_i = [1 + (7141 \times (i + 1) + 73)]\ mod\ (100)$	$i = 0, .., n,\ p_{n+1} = 0$
Generation 3:	$p_i = 1 + \lfloor \frac{99 \times t_{0i}}{\theta} \rfloor$	$i = 0, .., n,\ p_{n+1} = 0,$ $\theta = \max_{v \in V_0} t_{0v}$

Our detailed experimental results on these benchmark instances are presented in Appendix 2. The proposed GRASP-SR is compared with the TS and 2-PIA heuristics in Tables 7, 8 and 9 and is compared with GRASP and GRASP-PR heuristics in Tables 10, 11 and 12. In these tables, column BC corresponds to the exact Branch-and-Cut method proposed by Fischetti et al. (1998). The time limit of 5 h was considered for BC and for the instances that this time limit was exceeded before returning the optimal solution, the best available solution was reported. The value for running time of these instances was reported as “t.l.”.

Although our GRASP-SR method with the first configuration almost outperforms the solutions of other heuristic methods in the literature, it takes more time to reach its best result. Therefore, it was decided to consider our configuration with the time limit of 2 min to have a better comparison of the proposed method versus others. The results summary for this configuration is reported in Tables 4 and 5.

Each entry in the tables indicates the number of instances for which a better solution was found by the method associated with its row as compared with the method associated with its column. The row named Optimal corresponds to the number of instances for which each method could find the optimal solution or a solution equal or better than the best known solution. Finally, the last row, named optimal(5000), reports the number of instances for which our GRASP-SR method with the first configuration could find the optimal solution or a solution equal or better than the best known solution. As an example, considering the Generation 3 instances, the 2-PIA

Table 4 GRASP-SR in comparison with the TS and 2-PIA approaches on the TSP-based benchmark instances

	Generation-1 (42 instances)			Generation-2 (42 instances)			Generation-3 (42 instances)		
	TS	2-PIA	GRASP-SR	TS	2-PIA	GRASP-SR	TS	2-PIA	GRASP-SR
TS	–	3	1	–	4	1	–	8	4
2-PIA	23	–	1	28	–	3	31	–	6
GRASP-SR	28	22	–	33	27	–	33	22	–
Optimal	14	19	31	9	14	29	5	13	28
Optimal (5000)			38			29			31

Table 5 GRASP-SR in comparison with GRASP and GRASP-PR approaches on the TSP-based benchmark instances

	Generation-1 (40 instances)			Generation-2 (40 instances)			Generation-3 (40 instances)		
	GRASP	GRASP-PR	GRASP-SR	GRASP	GRASP-PR	GRASP-SR	GRASP	GRASP-PR	GRASP-SR
GRASP	–	0	1	–	0	1	–	0	1
GRASP-PR	17	–	2	29	–	3	34	–	5
GRASP-SR	19	7	–	31	22	–	33	17	–
Optimal	20	30	32	9	16	26	6	18	26
Optimal (5000)			36			29			28

outperforms our GRASP-SR in 6 instances, while GRASP-SR outperforms 2-PIA in 22 instances. Furthermore, 2-PIA and GRASP-SR found 13 and 28 optimal or best known solutions, respectively.

The results show that GRASP-SR almost outperforms all the mentioned state-of-the-art heuristics. GRASP-SR with the time limit of 2 min outperforms the TS, 2-PIA, GRASP and GRASP-PR in 94, 71, 83 and 46 instances, respectively and performs worse in only 6, 10, 3 and 10 instances. In other words, the method outperforms the TS, 2-PIA, GRASP, and GRASP-PR in 75, 56, 69 and 38 percent of the instances and performs worse than them in only 5, 8, 3, and 8 percent of the instances.

Moreover, the proposed algorithm could find the optimal or best known solution of about 70 % of the instances with only 2 min of computation time for solving each instance. The TS, the 2-PIA, GRASP and GRASP-PR could find about 22, 37, 29 and 53 percent of the optimal or best known solutions, respectively.

According to Table 8, for problem pr226 of the Generation 2 instances, Silberholz and Golden (2010) found a solution of 6641, better than the solution of 6615 obtained within 5 h of computation by Fischetti et al. (1998). GRASP-SR improved the solution to 6662 in less than a second. For problem pr299 of this generation, we have obtained a solution of 9173 in about 100 min, which is better than the solution of 9,161 obtained within 5 h of computation by Fischetti et al. (1998). For this instance, the proposed algorithm could also find a solution of 9165 in less than 2 min of computation time, which is also better than the best know solution for the problem. The routes corresponding to these new solutions are reported in the footnote of Table 8.

5 Conclusions

In this paper, a novel GRASP solution was proposed for the OP. Being able to find the optimal or best known solution of about 70 % of the benchmark instances, which is about 17 % more than the achievement of the best known heuristic approach in the literature, the proposed algorithm outperforms the state-of-the-art heuristic methods. In addition, the method improved the solution quality of two standard benchmark instances. In future studies, the proposed method is expected to be rather simply and effectively applicable to similar routing problems.

Acknowledgments The authors would like to thank the authors of Fischetti et al. (1998) and Campos et al. (2014) for sharing their results.

Appendix 1: GRASP-SR parameter selection

In the construction phase of GRASP-SR (Sect. 3.1), the candidate list was restricted to the paths having an improvement within the fraction $\alpha = 0.2$ of the profit gained through the most profitable path in the candidate list. Campos et al. (2014) presented some experiments to show that the value of 0.2 is a good choice for their work. In this appendix, we show that this value is also a good choice in this work.

We considered the 48 TSP-based problem instances of Fischetti et al. (1998) having no more than 100 vertices (16 instances in each Generation). GRASP-SR was run for

Table 6 Effect of different α values on the results of GRASP-SR

α	0.0	0.2	0.4	0.6	0.8	1.0
Dev.	3.583	**1.417**	1.625	6.75	12.292	71.146
Optimal	**41**	**41**	40	37	33	22

different values of α (0, 0.2, 0.4, 0.6, 0.8, 1.0) 100 times on each instance and the best solution of these 100 runs were kept for each instance. All the results were obtained in less than 2 min. For each value of the parameter α, Table 6 shows the average deviation of the solutions with respect to the optimal solutions (Dev.) and the number of optimal solutions that GRASP-SR with the given parameter has been able to find (Optimal).

The results show that when α is equal to 0 or 0.2, a larger number of instances can be solved to optimality. Additionally, when α is set to 0.2, the smallest average deviation of the results from the optimal solutions is obtained.

Appendix 2: Detailed results on the orienteering problem

In this section, detailed results for the TSP-based benchmark instances are presented (Tables 7, 8, 9, 10, 11 and 12). The description of the tables provided in this section was presented in Sect. 4. In addition, the values in bold indicate the best solution among the solutions of the reported exact and heuristic approaches.

Some additional notes should be considered. As Silberholz and Golden (2010) mentioned, herein the value of T_{max} has been corrected for problem gr229 to 67,301, which was incorrectly listed as 1765 in Fischetti et al. (1998). The correct value of T_{max} for problem lin318 is 21,015, but since the value of 21,045 has been used in other works, we also used this value for our experimental results. Moreover, the profit values produced by Campos et al. (2014) for the Generation 3 instances does not contain the profit value of 100 for the instances rat99, kroc100, kroe100, pr124, and krob150. Due to floating-point precision errors, for the farthest vertex from the origin of these instances, the value of 99 has been produced instead of 100. It influenced our results for instances rat99 and kroe100, resulting in a value one more than the optimal solutions reported by Campos et al. (2014). Therefore, the same profit values were used as used in Campos et al. (2014) for a fair comparison.

Table 7 Detailed results of GRASP-SR in comparison with the exact Branch-and-Cut (BC) (Fischetti et al. 1998), TS (Gendreau et al. 1998b), and 2-PIA (Silberholz and Golden 2010) approaches on Generation 1 of the TSP-based benchmark instances

Instance		BC		TS		2-PIA		GRASP-SR						
Name	T_{max}	Opt.	Sec.	Value	Sec.	Value	Sec.	2-min.	Best^n	Avg.	Worst^n	Sec.	$\#iter$	Size
att48	5314	**31**	0.7	**31**	0.96	**31**	0.34	**31**	**31**			0.00	1	32
gr48	2523	**31**	1.1	**31**	0.96	**31**	0.34	**31**	**31**			0.00	1	32
hk48	5731	**30**	1.7	**30**	0.87	**30**	0.46	**30**	**30**			0.06	35	31
eil51	213	**29**	1.2	**29**	0.96	**29**	0.54	**29**	**29**			0.05	11	30
brazil58	12698	**46**	3.2	45	1.39	**46**	0.46	**46**	**46**			0.16	46	47
st70	338	**43**	5.3	**43**	2.06	**43**	1.04	**43**	**43**			0.02	2	44
eil76	269	**47**	5.7	46	1.94	46	0.96	**47**	**47**			0.39	45	48
pr76	54080	**49**	50.9	**49**	2.10	**49**	1.00	**49**	**49**			0.02	1	50
gr96	27605	**64**	121.1	60	2.28	**64**	2.20	**64**	**64**			0.05	2	65
rat99	606	**52**	53.5	51	2.15	51	1.05	**52**	**52**			0.19	8	53
kroa100	10641	**56**	27.1	55	3.51	**56**	1.31	**56**	**56**			0.42	18	57
krob100	11071	**58**	326.9	**58**	3.25	**58**	1.64	**58**	**58**			0.78	49	59
kroc100	10375	**56**	50.7	54	2.56	55	1.71	**56**	**56**			0.45	31	57
krod100	10647	**59**	32.0	55	2.48	**59**	1.88	**59**	**59**			0.09	4	60
kroe100	11034	**57**	776.0	**57**	2.49	56	1.24	**57**	**57**			0.12	6	58
rd100	3955	**61**	30.2	60	2.75	**61**	1.76	**61**	**61**			0.31	12	62
eil101	315	**64**	7.1	62	2.90	**64**	1.28	**64**	**64**			0.23	10	65
lin105	7190	**66**	83.4	**66**	3.23	**66**	1.49	**66**	**66**			0.02	1	67
pr107	22152	**54**	86.3	**54**	1.43	**54**	0.90	**54**	**54**			0.02	2	55
gr120	3471	**75**	17.5	73	3.57	74	2.78	**75**	**75**			0.97	15	76
pr124	29515	**75**	41.2	**75**	4.06	**75**	1.54	**75**	**75**			0.17	10	76
bier127	59141	**103**	73.7	**103**	7.94	**103**	2.79	**103**	**103**			2.53	56	104

Table 7 continued

Instance		BC		TS		2-PIA		GRASP-SR						
Name	T_{max}	Opt.	Sec.	Value	Sec.	Value	Sec.	2-min.	Best^n	Avg.	Worst^n	Sec.	$\#iter$	Size
pr136	48386	**71**	214.2	68	2.89	69	2.16	**71**	**71**			0.42	5	72
gr137	34927	**81**	178.6	**81**	6.08	**81**	2.77	**81**	**81**			0.42	7	82
pr144	29269	**77**	240.3	73	2.81	73	2.20	**77**	**77**			0.02	1	78
kroa150	13262	**86**	4669.0	83	5.51	85	5.02	**86**	**86**			3.01	31	87
krob150	13065	**87**	145.6	80	4.88	86	4.50	**87**	**87**			11.45	137	88
pr152	36841	**77**	204.6	74	3.39	76	4.50	**77**	**77**			0.17	7	78
u159	21040	**93**	497.6	83	3.88	82	3.79	**93**	**93**			3.04	31	94
rat195	1162	**102**	331.9	97	7.81	98	4.99	101	**102**^{2}		101^8	953.60	2843	103
d198	7890	**123**	716.3	116	9.73	117	7.19	122	**123**^{9}		122^1	120.86	465	124
kroa200	14684	**117**	395.0	102	5.37	110	7.70	**117**	**117**			69.13	226	118
krob200	14719	**119**	683.6	109	9.36	116	8.24	**119**	**119**^{2}		118^8	66.11	246	120
gr202	20080	**147**	150.6	136	13.46	139	10.28	146	**147**^{1}		146^9	1854.22	4605	148
ts225	63322	**125**	t.l.	121	12.47	124	9.89	124	124			0.09	2	125
pr226	40185	**134**	t.l.	105	8.23	116	9.32	125	126^2		125^8	232.11	1338	127
gr229	67301	**176**	75.0	174	27.65	173	13.86	175	175^5		174^5	31.62	43	176
gil262	1189	**158**	120.6	139	13.65	147	18.32	155	**158**^{1}	156.2	155^1	1852.38	2195	159
pr264	24568	**132**	2860.2	**132**	13.53	**132**	7.36	109	**132**^{1}	113.6	109^1	1282.90	4011	133
pr299	24096	**162**	14244.0	152	19.36	154	36.76	160	**162**^{2}		161^8	3592.67	2152	163
lin318	21045	**205**	3169.9	176	33.52	194	34.42	204	**205**^{1}	203.8	203^3	5252.95	3094	206
rd400	7641	**239**	4272.5	216	46.27	218	99.27	231	235^2	233.6	233^6	3724.50	633	236

Table 8 Detailed results of GRASP-SR in comparison with the exact Branch-and-Cut (BC) (Fischetti et al. 1998), TS (Gendreau et al. 1998b), and 2-PIA (Silberholz and Golden 2010) approaches on Generation 2 of the TSP-based benchmark instances

Instance		BC		TS		2-PIA		GRASP-SR						
Name	T_{max}	Opt.	Sec.	Value	Sec.	Value	Sec.	2-min.	Best^n	Avg.	Worst^n	Sec.	$\#iter$	Size
att48	5314	**1717**	3.9	**1717**	0.97	**1717**	0.38	**1717**	**1717**			0.03	14	32
gr48	2523	**1761**	18.0	1749	0.98	1750	0.49	**1761**	**1761**^{8}		1750^{2}	0.08	50	28
hk48	5731	**1614**	7.1	**1614**	0.76	**1614**	0.57	**1614**	**1614**			0.08	23	28
eil51	213	**1674**	30.7	**1674**	0.85	**1674**	0.83	**1674**	**1674**			0.02	2	26
brazil58	12698	**2220**	7.8	2198	1.58	**2220**	0.60	**2220**	**2220**			1.84	365	43
st70	338	**2286**	181.0	2285	1.57	**2286**	1.43	**2286**	**2286**			1.40	164	40
eil76	269	**2550**	7.2	2490	1.64	2540	1.47	**2550**	**2550**			0.12	6	42
pr76	54080	**2708**	62.0	**2708**	2.04	**2708**	1.53	**2708**	**2708**			0.02	1	45
gr96	27605	**3425**	453.1	3156	2.32	3376	2.17	**3425**	**3425**			1.48	39	62
rat99	606	**2944**	125.4	2793	2.06	2926	2.52	**2944**	**2944**			0.11	3	47
kroa100	10641	**3212**	67.8	**3212**	2.96	**3212**	2.70	**3212**	**3212**			1.25	37	56
krob100	11071	**3241**	481.4	3217	2.18	3237	2.72	**3241**	**3241**			0.28	8	50
kroc100	10375	**2947**	316.2	2818	2.16	**2947**	2.60	**2947**	**2947**			0.28	10	49
krod100	10647	**3307**	334.2	3268	2.49	3299	2.74	**3307**	**3307**			2.28	72	55
kroe100	11034	**3090**	1433.9	3082	2.54	3009	2.80	**3090**	**3090**^{2}	3088.7	3084^{1}	107.16	3978	55
rd100	3955	**3359**	27.8	**3359**	2.47	**3359**	2.28	**3359**	**3359**			5.24	160	58
eil101	315	**3655**	296.5	3642	2.62	3634	3.23	**3655**	**3655**^{2}	3648.5	3643^{3}	60.84	1490	59
lin105	7190	**3544**	163.6	3536	3.69	3536	2.34	**3544**	**3544**			0.03	1	62
pr107	22152	**2667**	99.4	**2667**	1.60	**2667**	1.61	**2667**	**2667**			0.00	1	55
gr120	3471	**4371**	650.0	4355	4.02	4333	2.88	**4371**	**4371**^{8}		4370^{2}	36.21	568	70
pr124	29515	**3917**	79.4	**3917**	4.15	**3917**	3.08	**3917**	**3917**			0.66	19	75

Table 8 continued

Instance		BC		TS		2-PIA		GRASP-SR						
Name	T_{max}	Opt.	Sec.	value	Sec.	value	Sec.	2-min.	Best^n	Avg.	Worst^n	Sec.	$\#iter$	size
bier127	59141	**5383**	245.8	5309	6.37	5368	5.80	5373	5379^{2}	5362.3	5345^{1}	199.03	1522	97
pr136	48386	**4309**	194.3	4037	3.69	4234	5.31	**4309**	**4309**^{3}	4298.9	4287^{3}	29.80	393	66
gr137	34927	**4294**	3193.0	4274	4.95	4271	4.84	**4294**	**4294**^{5}	4292.8	4289^{1}	56.02	449	80
pr144	29269	**4003**	1409.0	3890	3.46	**4003**	3.75	**4003**	**4003**			4.85	117	75
kroa150	13262	**4918**	3950.6	4788	5.33	4903	7.79	4915	4915^{3}	4913.3	4912^{3}	92.59	564	83
krob150	13065	**4869**	1018.1	4587	4.68	4853	7.20	**4869**	**4869**^{5}	4861.6	4853^{1}	90.62	679	80
pr152	36841	**4279**	188.6	4130	3.13	4269	4.89	**4279**	**4279**^{2}		4274^{8}	20.70	360	74
u159	21040	**4960**	1772.4	4804	3.82	4863	5.77	**4960**	**4960**^{6}	4956.3	4950^{3}	2.23	19	87
rat195	1162	**5791**	2498.6	5415	5.48	5695	11.05	5755	5786^{1}	5770.4	5754^{1}	1767.34	3639	94
d198	7890	**6670**	2517.1	6530	8.62	6608	12.76	6650	6669^{1}	6662.5	6654^{1}	204.75	372	111
kroa200	14684	**6547**	805.1	6200	7.54	6436	17.10	6524	6544^{1}	6537.7	6534^{3}	3084.41	4459	110
krob200	14719	**6419**	3522.8	6098	9.07	6349	14.05	6339	6404^{1}	6384.7	6347^{1}	1675.66	3362	103
gr202	20080	**7848**	3847.6	7627	15.73	7760	15.11	7774	7809^{1}	7799.3	7790^{1}	732.95	812	132
ts225	63322	**6834**	1195.5	6491	12.90	6731	7.90	6779	6808^{2}	6790.8	6775^{1}	177.15	628	121
pr226^a	40185	6615	t.l.	6085	9.35	6641	12.63	**6662**	**6662**			0.69	2	117
gr229	67301	**9187**	4261.4	8965	24.37	9131	17.42	9109	9151^{1}	9131.9	9120^{1}	3796.78	2099	164
gil262	1189	**8321**	5574.6	7689	9.98	8144	37.96	8170	8286^{1}	8234.1	8217^{1}	5918.80	3285	136
pr264	24568	**6654**	4253.3	**6654**	13.60	**6654**	18.74	6392	6406^{1}	6375.2	6338^{1}	2358.51	4036	108

Table 8 continued

Instance		BC		TS		2-PIA		GRASP-SR						
Name	T_{max}	Opt.	Sec.	value	Sec.	value	Sec.	2-min.	Best^n	Avg.	Worst^n	Sec.	$\#iter$	size
pr299[b]	24096	9161	t.l.	8436	19.50	8938	33.03	9165	**9173**[1]	9161.6	9149[1]	6028.45	2511	146
lin318	21045	**10900**	t.l.	9233	30.04	10755	85.13	10766	10880[1]	10855.8	10816[1]	8572.52	2285	184
rd400	7641	**13648**	t.l.	12114	43.05	13120	149.47	13274	13529[1]	13471.7	13429[1]	46697.30	3154	213

[a] Route with a new profit 6662 : 0 1 2 3 4 5 7 10 15 66 64 51 52 53 54 55 56 57 58 59 60 61 62 94 86 88 95 87 89 90 98 99 100 92 93 136 137 138 144 146 139 148 141 142 149 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 124 131 129 122 121 120 119 127 125 109 117 116 115 107 106 105 103 112 111 104 102 110 85 84 83 82 81 80 79 78 77 76 75 74 73 0

[b] Route with a new profit 9173 : 0 3 2 1 5 4 7 8 9 10 11 12 13 87 88 90 85 86 93 95 94 97 98 83 80 81 78 19 17 20 22 23 24 25 71 72 73 70 68 66 33 32 31 34 35 37 36 41 39 40 42 44 45 46 43 56 59 60 110 112 122 125 124 176 171 170 178 188 186 251 254 260 259 261 263 264 266 267 248 243 244 246 192 193 195 241 242 239 236 237 271 272 273 234 276 275 232 229 227 199 198 200 165 164 130 129 103 128 131 132 134 136 137 161 298 226 225 222 203 204 205 210 209 207 219 220 221 211 212 217 215 286 287 288 290 289 214 154 153 152 151 150 149 148 146 0

Table 9 Detailed results of GRASP-SR in comparison with the exact Branch-and-Cut (BC) (Fischetti et al. 1998), TS (Gendreau et al. 1998b), and 2-PIA (Silberholz and Golden 2010) approaches on Generation 3 of the TSP-based benchmark instances

Instance		BC		TS		2-PIA		GRASP-SR						
Name	T_{max}	Opt.	Sec.	Value	Sec.	Value	Sec.	2-min.	Best^{n}	Avg.	Worst^{n}	Sec.	$\#iter$	Size
att48	5314	**1049**	251.8	1044	0.76	**1049**	0.44	**1049**	**1049**			0.02	17	30
gr48	2523	**1480**	27.5	1479	0.93	**1480**	0.36	**1480**	**1480**			0.00	2	32
hk48	5731	**1764**	3.5	1754	0.99	**1764**	0.40	**1764**	**1764**			0.02	1	29
eil51	213	**1399**	19.5	**1399**	0.96	**1399**	0.50	**1399**	**1399**			0.03	6	28
brazil58	12698	**1702**	2.8	**1702**	1.50	**1702**	0.69	**1702**	**1702**			0.11	29	43
st70	338	**2108**	70.6	2079	1.60	**2108**	1.09	**2108**	**2108**			0.06	8	37
eil76	269	**2467**	49.6	**2467**	2.08	2462	1.62	**2467**	$\mathbf{2467}^{7}$		2462^{3}	1.48	109	45
pr76	54080	**2430**	34.0	**2430**	2.08	**2430**	1.46	**2430**	$\mathbf{2430}^{6}$		2426^{4}	0.47	48	48
gr96	27605	**3182**	416.6	2965	2.59	3145	1.73	**3182**	**3182**			2.12	54	64
rat99	606	**2908**	487.4	2816	2.10	**2908**	1.81	**2908**	**2908**			0.17	8	46
kroa100	10641	**3211**	248.8	3188	3.10	**3211**	1.51	**3211**	**3211**			0.09	5	52
krob100	11071	**2804**	138.9	2754	2.02	**2804**	2.32	**2804**	**2804**			0.30	14	50
kroc100	10375	**3155**	228.1	3029	2.42	3149	1.63	**3155**	**3155**			0.11	3	53
krod100	10647	**3167**	230.3	3164	2.81	3123	2.40	**3167**	$\mathbf{3167}^{8}$	3163.5	3148^{1}	5.23	243	57
kroe100	11034	**3049**	184.4	3015	2.07	3021	2.75	**3049**	**3049**			0.05	2	46
rd100	3955	**2926**	1032.3	2792	2.61	2924	2.62	**2926**	$\mathbf{2926}^{9}$		2924^{1}	3.84	125	56
eil101	315	**3345**	186.7	3286	3.21	3335	2.36	**3345**	$\mathbf{3345}^{1}$		3335^{9}	63.63	1601	61
lin105	7190	**2986**	1121.1	2894	2.40	**2986**	2.04	**2986**	**2986**			1.62	74	59
pr107	22152	**1877**	17609.0	1756	1.57	**1877**	1.48	**1877**	**1877**			0.20	45	29
gr120	3471	**3779**	145.5	3673	3.56	3737	3.58	**3779**	$\mathbf{3779}^{5}$	3774.6	3759^{1}	24.87	315	70
pr124	29515	**3557**	11487.2	3439	3.28	3517	2.29	**3557**	$\mathbf{3557}^{3}$		3549^{7}	29.17	1742	72

Table 9 continued

Instance		BC		TS		2-PIA		GRASP-SR						
Name	T_{max}	Opt.	Sec.	value	Sec.	value	Sec.	2-min.	Best^n	Avg.	Worst^n	Sec.	#$iter$	size
bier127	59141	**2365**	2001.2	2351	5.45	2355	4.59	2356	2356^{1}	2342.8	2314^{1}	34.27	351	88
pr136	48386	**4390**	958.5	4132	3.60	**4390**	3.97	**4390**	**4390**			0.22	3	71
gr137	34927	**3979**	4958.7	3473	4.38	3947	4.81	**3979**	**3979**^{4}	3975.9	3973^{5}	101.06	1009	72
pr144	29269	**3809**	t.l.	3404	2.59	3634	3.74	3741	3741^{4}	3706	3661^{1}	3.49	146	62
kroa150	13262	**5039**	3828.9	**5039**	6.61	5037	5.97	**5039**	**5039**^{9}		5033^{1}	18.30	165	80
krob150	13065	**5314**	1363.9	4749	4.31	5229	5.47	**5314**	**5314**^{6}	5312.4	5309^{2}	107.31	529	86
pr152	36841	**3905**	13736.7	3896	3.62	3089	3.52	**3905**	**3905**^{4}	3867.3	3818^{3}	12.98	614	75
u159	21040	**5272**	1447.2	5056	6.95	5270	6.05	**5272**	**5272**			2.36	29	85
rat195	1162	**6195**	3975.4	5730	7.68	6140	9.91	6140	6191^{1}	6144.8	6121^{1}	1056.39	2906	94
d198	7890	**6320**	8635.7	6148	9.10	5950	9.68	6084	6163^{1}	6099.1	6042^{1}	1254.82	4621	116
kroa200	14684	**6123**	6548.9	6026	7.13	6047	9.23	6121	**6123**^{3}	6121.3	6118^{1}	139.84	262	104
krob200	14719	**6266**	783.7	6127	9.96	6240	10.22	6239	**6266**^{6}	6260.3	6239^{1}	249.96	634	103
gr202	20080	**8632**	11113.5	8271	14.64	8206	15.23	8248	8469^{1}	8384.2	8261^{1}	4817.08	4410	139
ts225	63322	**7575**	5821.8	7074	16.42	6954	11.46	**7575**	**7575**^{9}		7526^{1}	25.07	119	125
pr226	40185	**6993**	7923.2	4486	5.65	6665	16.30	6860	6912^{1}	6883.5	6859^{1}	660.90	4478	107
gr229	67301	**6347**	1891.5	6186	23.88	6225	18.55	6113	6235^{2}	6222.6	6181^{1}	1395.75	1118	144
gil262	1189	**9246**	9574.0	8638	14.58	9024	35.60	8992	9128^{1}	9095.4	9068^{1}	6579.07	3615	141
pr264	24568	**8137**	4011.3	4140	9.97	7828	23.08	7814	**8137**^{2}	8037.3	7850^{1}	385.48	683	106
pr299	24096	**10358**	t.l.	10057	16.68	9974	33.27	9953	10277^{1}	10147.2	9985^{1}	6295.96	3509	148
lin318	21045	**10382**	t.l.	9007	21.79	9876	66.29	10275	10275^{1}	10190.3	10132^{1}	61.09	15	181
rd400	7641	**13229**	t.l.	11689	36.98	12858	115.97	12794	13070^{1}	13010.1	12967^{1}	32744.80	3279	214

Table 10 Detailed results of GRASP-SR in comparison with the exact Branch-and-Cut (BC) (Fischetti et al. 1998), GRASP and GRASP-PR(Campos et al. 2014) approaches on Generation 1 of the TSP-based benchmark instances

Instance		BC		GRASP		GRASP-PR		GRASP-SR						
Name	T_{max}	Opt.	Sec.	Value	Sec.	Value	Sec.	2-min.	$Best^n$	Avg.	$Worst^n$	Sec.	$\#iter$	Size
att48	5314	**31**	0.7	**31**	0.1	**31**	1.5	**31**	**31**			0.0	1	32
gr48	2523	**31**	1.1	**31**	0.2	**31**	1.9	**31**	**31**			0.0	1	32
hk48	5731	**30**	1.7	**30**	0.2	**30**	3.2	**30**	**30**			0.1	35	31
eil51	213	**30**	1.2	**30**	0.2	**30**	0.3	**30**	**30**			0.0	7	31
brazil58	12698	**46**	3.2	**46**	0.3	**46**	12.4	**46**	**46**			0.1	46	47
st70	338	**44**	5.3	**44**	0.5	**44**	2.3	**44**	**44**			0.0	1	45
eil76	269	**47**	5.7	**47**	0.7	**47**	1.3	**47**	**47**			0.0	7	48
pr76	54080	**49**	50.9	48	0.6	**49**	44.1	**49**	**49**			0.0	1	50
gr96	27605	**64**	121.1	**64**	1.4	**64**	93.4	**64**	**64**			0.0	2	65
rat99	606	**53**	53.5	52	1.3	**53**	5.6	**53**	**53**			0.2	8	54
kroa100	10641	**57**	27.1	56	1.4	**57**	54.6	**57**	**57**			1.2	68	58
krob100	11071	**58**	326.9	**58**	1.4	**58**	54.4	**58**	**58**			1.1	59	59
kroc100	10375	**56**	50.7	**56**	1.3	**56**	40.5	**56**	**56**			1.4	91	57
krod100	10647	**59**	32.0	**59**	1.4	**59**	66.0	**59**	**59**			0.0	2	60
kroe100	11034	**57**	776.0	**57**	1.4	**57**	54.4	**57**	**57**			0.0	1	58
rd100	3955	**61**	30.2	**61**	1.7	**61**	44.3	**61**	**61**			0.2	8	62
eil101	315	**66**	7.1	65	1.7	**66**	3.6	**66**	**66**			3.4	131	67
lin105	7190	**66**	83.4	**66**	1.6	**66**	141.4	**66**	**66**			0.0	2	67
pr107	22152	**54**	86.3	**54**	0.6	**54**	25.1	**54**	**54**			0.0	3	55
gr120	3471	**75**	17.5	74	3.3	**75**	74.1	**75**	**75**			1.0	15	76
pr124	29515	**75**	41.2	**75**	2.5	**75**	145.2	**75**	**75**			0.1	5	76

Table 10 continued

Instance		BC		GRASP		GRASP-PR		GRASP-SR						
Name	T_{max}	Opt.	Sec.	value	Sec.	value	Sec.	2-min.	Best^n	Avg.	Worst^n	Sec.	$\#iter$	size
bier127	59141	**103**	73.7	102	3.5	**103**	213.5	**103**	**103**			0.3	6	104
pr136	48386	**71**	214.2	70	2.9	70	50.2	**71**	**71**			1.3	16	72
gr137	34927	**81**	178.6	**81**	4.5	**81**	380.6	**81**	**81**			0.4	7	82
pr144	29269	**77**	240.3	**77**	3.1	**77**	311.3	**77**	**77**			0.0	1	78
kroa150	13262	**86**	4669.0	85	5.1	**86**	94.5	**86**	**86**			0.0	1	87
krob150	13065	**87**	145.6	86	5.7	**87**	107.4	**87**	**87**			0.9	11	88
pr152	36841	**77**	204.6	**77**	3.1	**77**	365.3	**77**	**77**			0.7	21	78
u159	21040	**93**	497.6	91	6.2	**93**	242.5	**93**	**93**			0.6	4	94
rat195	1162	**104**	331.9	101	11.7	102	36.0	103	**104**^{1}		103^{9}	218.5	653	105
d198	7890	**124**	716.3	122	15.1	123	424.1	**124**	**124**			24.5	107	125
kroa200	14684	**118**	395.0	116	16.0	117	245.7	**118**	**118**^{1}		117^{9}	9.8	31	119
krob200	14719	**119**	683.6	116	14.0	118	321.6	118	**119**^{6}		118^{4}	147.1	566	120
ts225	63322	**126**	t.l.	124	17.9	124	146.9	124	124			0.1	2	125
pr226	40185	**126**	t.l.	124	14.2	**126**	902.2	**126**	**126**^{4}		125^{6}	82.3	470	127
gil262	1189	**164**	120.6	158	34.5	160	183.2	159	163^{2}	161.6	160^{1}	1371.9	1783	164
pr264	24568	**132**	2860.2	**132**	11.8	**132**	1267.9	111	112^{3}	111.1	109^{1}	374.0	1161	113
pr299	24096	**162**	14224.0	156	63.2	158	933.3	161	**162**^{5}		161^{5}	1080.9	657	163
lin318	21045	**206**	3169.9	196	82.8	201	1703.7	202	**206**^{1}	204.3	203^{1}	234.6	140	207
rd400	7641	**243**	4272.5	228	176.2	228	444.2	234	238^{1}	236.7	236^{4}	4188.5	704	239

Table 11 Detailed results of GRASP-SR in comparison with the exact Branch-and-Cut (BC) (Fischetti et al. 1998), GRASP and GRASP-PR(Campos et al. 2014) approaches on Generation 2 of the TSP-based benchmark instances

Instance		BC		GRASP		GRASP-PR		GRASP-SR						
Name	T_{max}	Opt.	Sec.	Value	Sec.	Value	Sec.	2-min.	Best^n	Avg.	Worst^n	Sec.	$\#iter$	Size
att48	5314	**1634**	3.9	**1634**	0.1	**1634**	2.6	**1634**	**1634**			0.0	8	31
gr48	2523	**1469**	18.0	**1469**	0.1	**1469**	3.6	**1469**	**1469**			0.0	6	30
hk48	5731	**1535**	7.1	1524	0.1	**1535**	3.2	**1535**	**1535**			0.0	14	28
eil51	213	**1778**	30.7	**1778**	0.1	**1778**	3.8	**1778**	**1778**			0.0	10	28
brazil58	12698	**2326**	7.8	**2326**	0.2	**2326**	13.1	**2326**	**2326**			0.0	4	42
st70	338	**2302**	181.0	2294	0.3	**2302**	13.9	**2302**	**2302**			0.1	13	41
eil76	269	**2591**	7.2	2525	0.4	**2591**	12.0	**2591**	**2591**			0.0	2	43
pr76	54080	**2666**	62.0	2626	0.3	**2666**	14.6	**2666**	**2666**			0.0	2	46
gr96	27605	**3506**	453.1	3447	0.7	3501	35.8	**3506**	**3506**			0.5	21	58
rat99	606	**3042**	125.4	2976	0.7	3031	21.7	**3042**	$\mathbf{3042}^9$		3031^1	0.9	27	50
kroa100	10641	**3181**	67.8	3135	0.8	**3181**	29.1	**3181**	**3181**			0.1	5	53
krob100	11071	**3195**	481.4	3183	0.7	3191	31.4	3190	3190^8		3189^2	10.9	406	54
kroc100	10375	**3044**	316.2	**3044**	0.7	**3044**	21.7	**3044**	**3044**			0.1	4	47
krod100	10647	**3226**	334.2	3152	0.7	3212	33.1	**3226**	**3226**			6.8	179	55
kroe100	11034	**3310**	1433.9	3260	0.7	**3310**	36.5	**3310**	**3310**			0.2	7	55
rd100	3955	**3470**	27.8	3449	0.7	3453	33.7	**3470**	**3470**			0.1	3	55
eil101	315	**3668**	296.5	3596	0.8	3645	28.3	**3668**	**3668**			0.1	1	60
lin105	7190	**3577**	163.6	**3577**	0.9	**3577**	83.1	**3577**	**3577**			0.1	3	59
pr107	22152	**2681**	99.4	**2681**	0.5	**2681**	20.5	**2681**	**2681**			0.0	2	55
gr120	3471	**4223**	650.0	4138	1.3	4201	43.9	**4223**	**4223**			13.4	208	68
pr124	29515	**3840**	79.4	**3840**	1.3	**3840**	83.3	**3840**	$\mathbf{3840}^9$		3835^1	0.8	27	75

Table 11 continued

Instance		BC		GRASP		GRASP-PR		GRASP-SR						
Name	T_{max}	Opt.	Sec.	Value	Sec.	Value	Sec.	2-min.	Best^n	Avg.	Worst^n	Sec.	#$iter$	Size
bier127	59141	**5376**	245.0	5154	1.6	5264	100.6	5368	5368^{2}	5358.8	5345^{1}	20.5	201	100
pr136	48386	**4223**	194.3	4170	1.7	4213	36.8	**4223**	**4223**			0.3	5	66
gr137	34927	**4291**	3193.0	4255	1.9	4284	126.9	4282	4282^{2}	4258.1	4184^{1}	17.2	109	81
pr144	29269	**3994**	1409.0	3902	1.8	**3994**	114.9	**3994**	**3994**			8.4	177	75
kroa150	13262	**4919**	3950.6	4768	2.4	4915	52.8	**4919**	**4919**			1.7	13	81
krob150	13065	**5017**	1018.1	4967	2.2	5001	66.5	**5017**	**5017**			0.2	2	79
pr152	36841	**4196**	188.6	4094	1.8	4175	166.7	**4196**	$\mathbf{4196}^{5}$	4194.7	4193^{4}	100.6	1810	74
u159	21040	**5044**	1772.4	4809	2.8	4987	100.1	5034	$\mathbf{5044}^{2}$	5036.4	5032^{2}	118.3	816	86
rat195	1162	**5936**	2498.6	5693	5.3	5693	63.8	5893	5922^{3}	5909.8	5899^{2}	158.1	338	94
d198	7890	**6539**	2517.1	6347	5.8	6476	241.6	6511	6531^{1}	6519.9	6511^{1}	358.0	849	113
kroa200	14684	**6616**	805.1	6447	6.1	6551	111.5	6577	$\mathbf{6616}^{2}$	6602.8	6580^{1}	1753.8	3054	110
krob200	14719	**6597**	3522.8	6357	5.6	6409	103.9	6593	6594^{6}	6592.8	6589^{1}	159.4	328	110
ts225	63322	**6812**	1195.5	6701	8.9	6784	93.6	6791	$\mathbf{6812}^{4}$	6806.9	6798^{1}	366.8	1319	123
pr226	40185	**6691**	t.l.	6375	6.9	6614	265.5	**6691**	**6691**			27.3	98	121
gil262	1189	**9159**	5574.6	8847	14.6	8941	132.8	9055	9132^{1}	9120.1	9098^{1}	6160.7	4268	150
pr264	24568	**6666**	4253.3	**6666**	9.6	**6666**	343.7	6409	6422^{1}		6409^{9}	1542.5	2797	102
pr299	24096	**9107**	t.l.	8645	19.7	8689	400.9	8923	9096^{1}	9026.5	8986^{1}	9647.3	3836	144
lin318	21045	**10962**	t.l.	10074	27.3	10339	339.7	10799	10947^{2}	10919.2	10876^{1}	557.0	143	195
rd400	7641	**13555**	t.l.	12365	54.0	12365	229.2	13197	13465^{1}	13372	13320^{1}	34693.1	2732	211

Table 12 Detailed results of GRASP-SR in comparison with the exact Branch-and-Cut (BC) (Fischetti et al. 1998), GRASP and GRASP-PR(Campos et al. 2014) approaches on Generation 3 of the TSP-based benchmark instances

Instance		BC		GRASP		GRASP-PR		GRASP-SR						
Name	T_{max}	Opt.	Sec.	value	Sec.	value	Sec.	2-min.	Best^{n}	Avg.	Worst^{n}	Sec.	$\#iter$	size
att48	5314	**1050**	251.8	**1050**	0.1	**1050**	4.0	**1050**	**1050**			0.0	17	30
gr48	2523	**1481**	27.5	**1481**	0.1	**1481**	4.5	**1481**	**1481**			0.0	2	32
hk48	5731	**1765**	3.5	**1765**	0.1	**1765**	2.7	**1765**	**1765**			0.0	1	29
eil51	213	**1465**	19.5	**1465**	0.1	**1465**	5.3	**1465**	**1465**			0.1	18	30
brazil58	12698	**1703**	2.8	**1703**	0.2	**1703**	12.8	**1703**	**1703**			0.1	29	43
st70	338	**2182**	70.6	**2182**	0.3	**2182**	14.6	**2182**	**2182**			0.0	3	39
eil76	269	**2526**	49.6	2469	0.4	2523	17.5	**2526**	**2526**^{8}		2523^{2}	1.9	140	46
pr76	54080	**2431**	34.0	2421	0.4	**2431**	25.4	**2431**	**2431**^{9}		2421^{1}	4.8	489	48
gr96	27605	**3183**	416.6	3165	0.9	**3183**	52.5	**3183**	**3183**			2.0	54	64
rat99	606	**2950**	487.4	2929	0.9	2949	38.3	**2950**	**2950**			0.2	8	48
kroa100	10641	**3227**	248.8	3181	0.7	**3227**	51.1	**3227**	**3227**			0.1	3	51
krob100	11071	**2807**	138.9	2762	0.7	**2807**	61.0	**2807**	**2807**			0.0	3	50
kroc100	10375	**3158**	228.1	3114	0.8	3152	40.7	**3158**	**3158**			0.4	13	54
krod100	10647	**3173**	230.3	3168	0.8	**3173**	60.0	**3173**	**3173**^{4}	3171.5	3168^{1}	11.5	567	57
kroe100	11034	**3053**	184.4	3015	0.8	**3053**	69.6	**3053**	**3053**			0.0	2	48
rd100	3955	**2957**	1032.3	2940	0.9	2943	40.0	**2957**	**2957**			3.2	115	55
eil101	315	**3427**	186.7	3341	0.9	3389	29.5	**3427**	**3427**^{5}	3422.8	3415^{1}	61.1	1352	63
lin105	7190	**2995**	1121.1	2942	1.2	**2995**	101.7	**2995**	**2995**			0.0	2	60
pr107	22152	**1878**	17609.0	1876	0.6	**1878**	47.7	**1878**	**1878**			0.2	42	29
gr120	3471	**3780**	145.5	3687	1.7	3753	56.1	**3780**	**3780**^{5}	3775.6	3760^{1}	25.2	315	70
pr124	29515	**3558**	11487.2	3547	1.3	**3558**	98.5	**3558**	**3558**^{7}	3555.3	3547^{1}	1.5	87	72

Table 12 continued

Instance		BC		GRASP		GRASP-PR		GRASP-SR						
Name	T_{max}	Opt.	Sec.	value	Sec.	value	Sec.	2-min.	Best^{n}	Avg.	Worst^{n}	Sec.	$\#iter$	size
bier127	59141	**2366**	2001.2	2242	1.6	2336	116.5	2352	2358^{1}	2339.6	2311^{1}	385.5	4024	87
pr136	48386	**4391**	958.5	4363	2.1	4381	124.4	**4391**	**4391**			0.6	9	71
gr137	34927	**3980**	4958.7	3844	2.2	3946	100.4	**3980**	**3980**^{4}	3976.9	3974^{5}	100.9	1009	72
pr144	29269	**3747**	t.l.	3688	1.9	**3747**	176.6	3746	3746^{5}		3688^{5}	34.4	1416	62
kroa150	13262	**5050**	3828.9	5006	2.5	5043	157.1	**5050**	**5050**^{9}		5048^{1}	25.4	217	78
krob150	13065	**5337**	1363.9	5203	2.4	5318	113.0	5324	**5337**^{3}		5324^{7}	453.0	2259	87
pr152	36841	**3910**	13736.7	3871	2.3	**3910**	238.2	**3910**	**3910**^{3}		3822^{7}	0.5	24	76
u159	21040	**5273**	1447.2	5198	3.2	**5273**	232.0	**5273**	**5273**			1.2	16	85
rat195	1162	**6296**	3975.4	5970	7.2	6164	124.7	6244	6257^{2}	6240.3	6209^{1}	831.2	2237	94
d198	7890	**6369**	8635.7	6136	8.8	6222	384.7	6160	6243^{1}	6179.2	6092^{1}	253.9	928	118
kroa200	14684	**6140**	6548.9	5989	5.9	6089	205.1	6134	6138^{6}	6137	6134^{1}	987.2	1902	105
krob200	14719	**6274**	783.7	6024	5.2	6213	254.0	6269	**6274**^{5}	6270.7	6261^{1}	289.1	754	104
ts225	63322	**7618**	5821.8	7477	11.8	7576	263.2	7576	7576			27.3	129	125
pr226	40185	**6994**	7923.2	6802	7.0	6971	816.1	6876	6915^{1}	6877	6858^{1}	420.3	2875	107
gil262	1189	**9547**	9574.0	8911	15.9	9277	244.5	9389	9427^{1}	9405	9377^{1}	3097.5	1938	148
pr264	24568	**8138**	4011.3	7752	12.8	8013	901.2	7736	8106^{2}	8002.3	7854^{1}	2229.6	3977	103
pr299	24096	**10356**	t.l.	9848	26.0	10145	959.9	9963	10343^{2}	10154.3	9991^{1}	342.1	175	149
lin318	21045	**10430**	t.l.	9664	29.1	9876	434.9	10102	10270^{1}	10228	10155^{1}	13000.4	4147	180
rd400	7641	**13422**	t.l.	12681	56.4	12743	512.6	13080	13238^{1}	13184.7	13144^{1}	24406.5	2577	214

References

Archetti, C., Speranza, M.G., Vigo, D.: Vehicle Routing Problems with Profits. In: Toth, P., Vigo, D. (eds.) Vehicle Routing: Problems, Methods and Applications, MOS-SIAM Series on Optimization. SIAM, Philadelphia (2014)

Campos, V., Martí, R., Sánchez-Oro, J., Duarte, A.: GRASP with path relinking for the orienteering problem. J. Oper. Res. Soc. **65**(12), 1800–1813 (2014)

Chao, I.M., Golden, B.L., Wasil, E.A.: A fast and effective heuristic for the orienteering problem. Eur. J. Oper. Res. **88**(3), 475–489 (1996a)

Chao, I.M., Golden, B.L., Wasil, E.A.: The team orienteering problem. Eur. J. Oper. Res. **88**(3), 464–474 (1996b)

Feo, T., Resende, M.: Greedy randomized adaptive search procedures. J. Glob. Optim. **6**(2), 109–133 (1995)

Fischetti, M., González, J.J.S., Toth, P.: Solving the orienteering problem through branch-and-cut. INFORMS J. Comput. **10**(2), 133–148 (1998)

Gavalas, D., Konstantopoulos, C., Mastakas, K., Pantziou, G.: A survey on algorithmic approaches for solving tourist trip design problems. J. Heuristics **20**(3), 291–328 (2014)

Gendreau, M., Laporte, G., Semet, F.: A branch-and-cut algorithm for the undirected selective traveling salesman problem. Networks **32**(4), 263–273 (1998a)

Gendreau, M., Laporte, G., Semet, F.: A tabu search heuristic for the undirected selective travelling salesman problem. Eur. J. Oper. Res. **106**(23), 539–545 (1998b)

Laporte, G., Martello, S.: The selective travelling salesman problem. Discr. Appl. Math. **26**(23), 193–207 (1990)

Ramesh, R., Yoon, Y.S., Karwan, M.H.: An optimal algorithm for the orienteering tour problem. ORSA J. Comput. **4**(2), 155–165 (1992)

Reinelt, G.: TSPLIB—a traveling salesman problem library. ORSA J. Comput. **3**(4), 376–384 (1991)

Schilde, M., Doerner, K., Hartl, R., Kiechle, G.: Metaheuristics for the bi-objective orienteering problem. Swarm Intell. **3**(3), 179–201 (2009)

Silberholz, J., Golden, B.: The effective application of a new approach to the generalized orienteering problem. J. Heuristics **16**(3), 393–415 (2010)

Tang, H., Miller-Hooks, E.: A tabu search heuristic for the team orienteering problem. Comput. Oper. Res. **32**(6), 1379–1407 (2005)

Tsiligirides, T.: Heuristic methods applied to orienteering. J. Oper. Res. Soc. **35**(9), 797–809 (1984)

Vansteenwegen, P., Souffriau, W., Vanden Berghe, G., Van Oudheusden, D.: Metaheuristics for tourist trip planning. In: Sörensen, K., Sevaux, M., Habenicht, W., Geiger, M.J. (eds.) Metaheuristics in the Service Industry. Lecture Notes in Economics and Mathematical Systems, vol. 624, pp. 15–31. Springer, Berlin (2009)

Vansteenwegen, P., Souffriau, W., Van Oudheusden, D.: The orienteering problem: a survey. Eur. J. Oper. Res. **209**(1), 1–10 (2011)